

The auto-pst-pdf package

Will Robertson & Johannes Große

wspr 81 at gmail dot com

2008/03/14 v0.5

1 Basic usage

This package provides a wrapper around pst-pdf to automatically accomodate for typesetting either with DVI or PDF output. With default package option [on], typesetting under pdfL^AT_EX will automatically initiate an auxiliary compilation of L^AT_EX → dvips → ps2pdf → pdfcrop to generate the required PDF figures for the document.

After this has been done and the figures no longer need to be re-generated, the package can be given the [off] option to save compilation time:

```
\usepackage[off]{auto-pst-pdf}
```

If the extension of your L^AT_EX document is not .tex, then it must be declared when the package is loaded (*e.g.*, like many others, I like to use .ltx to distinguish between Plain T_EX and L^AT_EX files):

```
\usepackage[ext=ltx]{auto-pst-pdf}
```

2 Requirements

pdfT_EX must be called with the -shell-escape option. Requires the following packages: ifplatform, pst-pdf, xkeyval.

Heiko Oberdiek's pdfcrop Perl script¹ must be installed for the default `crop=on` option (see section 4). Under Windows, a Perl installation² will also need to be installed even though pdfcrop itself is part of MiK_TE_X.

¹<http://www.ctan.org/tex-archive/support/pdfcrop/>

²Freely available: <http://www.activestate.com/Products/activeperl/index.plex>

3 Provided macros for including graphics

Macros are provided to easily facilitate figures created by the MATLAB package `laprint`³ and the Mathematica package `MathPSfrag`⁴. Also, a generic `psfrag`⁵ wrapper is provided.

<code>\mathfig{<filename>}</code>	insert a Mathematica graphic from <code>MathPSfrag</code> (without <code>-psfrag</code> suffix)
<code>\matlabfig{<filename>}</code>	insert a MATLAB graphic from <code>laprint</code>
<code>\psfragfig{<filename>}</code>	insert an EPS with <code>psfrag</code>

The above commands all accept an optional argument which is passed to the underlying `\includegraphics` macro.

The `\matlabfig` command meddles slightly with the output of `laprint`; the font sizes in the figure will always be as originally defined. (This is unavoidable I'm afraid.)

For the `\psfragfig` command, `psfrag` statements are input from either or both of the files `<document>-psfrag.tex` and `<filename>-psfrag.tex` if they exist. Furthermore, supplementary `\psfrag` statements can be added in a trailing optional argument:

```
\psfrag[<graphics options>]{<filename>}[<psfrag statements>]
```

Manual `\psfrags` override those in `<filename>-psfrag.tex` which in turn override those in `<document>-psfrag.tex`.

4 Advanced package options

Better results are obtained by using `pdfcrop` in the auxiliary compilation process, and this is used by default. It is not installed by default, however, and will not always be required. Cropping can be controlled with the `crop` option:

```
\usepackage[crop=off]{auto-pst-pdf}
```

The package automatically deletes the files generated during the auxiliary \LaTeX compilation. Which files are deleted are chosen by passing a list of file extensions to the `cleanup` option (no error message or warning is produced if a file is specified that does not exist). The default list is:

```
\usepackage[cleanup={log,aux,dvi,ps,pdf}]{auto-pst-pdf}
```

The options passed individually to `latex`, `dvips`, `ps2pdf`, and `pdfcrop` in the auxiliary compilation process may all be customised, if you know what you're doing. The defaults for the latter three are

³<http://www.uni-kassel.de/fb16/rat/matlab/laprint/>

⁴<http://wwwth.mppmu.mpg.de/members/jgrosse/mathpsfrag/>

⁵<http://www.ctan.org/tex-archive/help/Catalogue/entries/psfrag.html>

```
\usepackage[dvips={-o -Ppdf},  
            pspdf={-dAutoRotatePages=/None}},  
            pdfcrop={}]{auto-pst-pdf}
```

The L^AT_EX auxiliary compilation has some hard-coded options (see the source if you're interested), and further options can be appended if you wish. For example, to run the auxiliary compilation with more information written to the console, use the following package option:

```
\usepackage[latex={-interaction=nonstopmode}]{auto-pst-pdf}
```

5 Acknowledgements

Many thanks to the authors of pst-pdf, psfrag, laprint, MathPSfrag, and pdfcrop. This package could not exist without their combined efforts over many years. Finally, Gernot HASSENFLUG deserves special mention for extensive testing, feature suggestions, and moral support :) Thanks, mate.

File I

auto-pst-pdf implementation

6 Setup code

This is the package.

```
1 \ProvidesPackage{auto-pst-pdf}[2008/03/14 v0.5 Wrapper for pst-pdf]
```

Change History

```
vo.3
  General: Too many changes to list. Command execution totally re-written. 4
vo.4
  General: Johannes tinkered with the code. Will will improve. :-)      4
  Will sorted it all out.                                                4
vo.5
  General: Removed \ifdefined to avoid e-TeX.                            8
  Removed mucking about with image extensions.                          7
  \app@convert: Fix PackageError (should have been a warning).          6
  \matlabfig: Redefine \resizebox for laprint.                           8
  \psfragfig: Extend \psfragfig to accept arbitrary input for extra \psfrag 9
  commands.
```

Required packages pst-pdf is loaded later on.

```
2 \RequirePackage{ifpdf,xkeyval,ifplatform}
```

Things we need

```
3 \newif\if@app@off@
4 \newif\if@app@crop@
5 \def\app@suffix{autopp}
6 \edef\app@jobname{\jobname-\app@suffix}
7 \edef\app@pics{\jobname-pics.pdf}
```

Option processing

```
8 \DeclareOptionX{off}[]{\@app@off@true}
9 \define@choicekey{auto-pst-pdf.sty}{crop}[\@tempa\@tempb]{on,off}{%
10 \ifcase\@tempb\relax
11 \@app@crop@true
12 \or
13 \@app@crop@false
```

```

14 \fi}
15 \DeclareOptionX{on}[]{\@app@off@false}
16 \DeclareOptionX{ext}{\def\app@ext{#1}}
17 \DeclareOptionX{latex}{%
18 \def\app@latex@opts{%
19 \ifwindows
20 -disable-write18
21 \else
22 -no-shell-escape
23 \fi
24 -jobname="\app@jobname"
25 -interaction=batchmode
26 #1}}
27 \DeclareOptionX{dvips}{\def\app@dvips@opts{#1}}
28 \DeclareOptionX{pspdf}{\def\app@pspdf@opts{#1}}
29 \DeclareOptionX{pdfcrop}{\def\app@pdfcrop@opts{#1}}
30 \DeclareOptionX{cleanup}{%
31 \let\app@rm@files\@empty
32 \@for\@ii:=#1\do{%
33 \edef\app@rm@files{\app@rm@files,\app@jobname.\@ii}}
34 \ExecuteOptionsX{%
35 ext=tex,
36 crop=on,
37 latex={},
38 dvips={-Ppdf},
39 pspdf={},
40 pdfcrop={},
41 cleanup={log,aux,dvi,ps,pdf}}
42 \ifwindows
43 \ExecuteOptionsX{pspdf={}}
44 \else
45 \ExecuteOptionsX{pspdf={-dAutoRotatePages=/None}}
46 \fi
47 \ProcessOptionsX

```

Shorthands

```

48 \def\app@exe{\immediate\write18}
49 \def\app@nl{^^J\space\space\space\space}
50 \newcommand\app@PackageError[2]{%
51 \PackageError{auto-pst-pdf}{\app@nl #1^^J}{#2}}
52 \newcommand\app@PackageWarning[1]{%
53 \PackageWarning{auto-pst-pdf}{\app@nl #1^^JThis warning occurred}}
54 \newcommand\app@PackageInfo[1]{\PackageInfo{auto-pst-pdf}{#1}}

```

These are cute:

```

55 \newcommand\OnlyIfFileExists[2]{\IfFileExists{#1}{#2}{}}
56 \newcommand\NotIfFileExists[2]{\IfFileExists{#1}{#2}{}}

```

`\app@convert` #1 : command name
 #2 : source file
 #3 : destination file
 Check if the source file exists and calls the command to generate the destination file. If the final file is not created, generate an error.

```

57 \def\app@convert#1#2#3{%
58   \OnlyIfFileExists{#2}{%
59     \app@exe{\csname app@cmd@#1\endcsname{#2}{#3}}%
60     \NotIfFileExists{#3}{\app@PackageWarning{Creation of #3 failed.}}}

```

`\app@compile` First we define the entire latex \rightarrow dvips \rightarrow ps2pdf (\rightarrow pdfcrop) command sequence. The actual call to the compilation macro follows thereafter. This macro contains the actual creation of the pdf container. Each processing step is in a separate macro to allow simple modification.

```

61 \def\app@compile{%
62   \app@cleanup
63   \app@remove@container
64   \app@convert{latex}{\jobname.\app@ext}{\app@jobname.dvi}%
65   \app@convert{dvips}{\app@jobname.dvi}{\app@jobname.ps}%
66   \if@app@crop@
67     \app@convert{pstopdf}{\app@jobname.ps}{\app@jobname.pdf}%
68     \app@convert{pdfcrop}{\app@jobname.pdf}{\app@pics}%
69   \else
70     \app@convert{pstopdf}{\app@jobname.ps}{\app@pics}%
71   \fi
72   \IfFileExists{\app@pics}
73   {\app@cleanup}
74   {\app@PackageWarning{Could not create \app@pics.
75     Auxiliary files not deleted.}}}

```

Command-line program to delete files:

```

76 \edef\app@rm{\ifwindows del \else rm -- \fi}

```

`\app@try@rm` Macro to delete files (comma-separated) if they exist:

```

77 \newcommand\app@try@rm[1]{%
78   \@for\@tempa:=#1\do{%
79     \OnlyIfFileExists{\@tempa}{\app@exe{\app@rm "\@tempa"}}}}

```

Remove pdf picture container:

```

80 \def\app@remove@container{\app@try@rm{\app@pics}}

```

Clean up auxiliary files: (`\app@rm@files` defined by the `cleanup` package option)

```
81 \def\app@cleanup{\app@try@rm{\app@rm@files}}
```

LaTeX:

```
82 \def\app@cmd@latex#1#2{latex \app@latex@opts\space
```

```
83 "\let\noexpand\APPmakepictures\noexpand\empty\noexpand\input #1"}
```

dvips:

```
84 \def\app@cmd@dvips#1#2{dvips \app@dvips@opts\space -o "#2" "#1"}
```

ps2pdf:

```
85 \def\app@cmd@pstopdf#1#2{ps2pdf \app@pspdf@opts\space "#1" "#2"}
```

pdfcrop:

```
86 \def\app@cmd@pdfcrop#1#2{pdfcrop \app@pdfcrop@opts\space "#1" "#2"}
```

6.1 Base functionality

For compilation, we use the `[notightpage]` option of `pst-pdf` and the `pdfcrop` Perl script because Matlab-created EPS figures (can) have elements that extend outside their bounding boxes, and end up with clipped content after `ps2pdf`. Otherwise the script `ps4pdf` would be sufficient.

pdfLaTeX compilation Requires supplementary processing with `pst-pdf`:

```
87 \ifpdf
88 \if@app@off@else
89 \ifshellescape
90 \app@exe{echo " "}
91 \app@exe{echo "-----"}
92 \app@exe{echo "auto-pst-pdf: Auxiliary LaTeX compilation"}
93 \app@exe{echo "-----"}
94 \app@compile
95 \app@exe{echo "-----"}
96 \app@exe{echo "auto-pst-pdf: End auxiliary LaTeX compilation"}
97 \app@exe{echo "-----"}
98 \else
99 \app@PackageError{%
100 "shell escape" (or "write18") is not enabled:\app@nl
101 auto-pst-pdf will not work!}
102 {You need to run LaTeX with the equivalent of
103 "pdflatex -shell-escape"\app@nl
104 Or turn off auto-pst-pdf.}%
105 \fi
106 \fi
107 \if@app@crop@
```

```

108     \PassOptionsToPackage{notightpage}{pst-pdf}
109 \fi

```

L^AT_EX compilation Either we’re calling latex from within a pdfL^AT_EX run (see above) or the document is being compiled as usual.

```

110 \else

```

L^AT_EX compilation from scratch (as in ‘`latex <document>.tex`’) — here the `postscript` environment does nothing and document is processed ‘normally’:

```

111     \ifx\APPmakepictures\undefined
112     \PassOptionsToPackage{inactive}{pst-pdf}

```

L^AT_EX compilation induced by this package:

```

113 \else
114     \if@app@crop@
115     \PassOptionsToPackage{notightpage}{pst-pdf}
116 \fi
117 \fi
118 \fi

```

After the requisite package options have been declared depending on the execution mode, it’s now time to load the package:

```

119 \RequirePackage{pst-pdf}

```

6.2 Extras for external packages

Commands are provided that mirror `\includegraphics` (and similarly accept an optional argument) for the output of different psfrag-related packages. This provides a consistent and easy way to include such figures in the document.

Please suggest wrappers for other packages that output psfrag figures (for example: SciLab, R, Maple, LabView, Sage, ... ?)

`\matlabfig` We need to disable the scaling that laprint applies to `\includegraphics` in here, because otherwise labels that extend outside the bounding box of the generated PostScript file will change the intended width of the graphic.

```

120 \let\app@ig\includegraphics
121 \newcommand\matlabfig[2] [] {%
122     \begin{postscript}
123     \renewcommand\resizebox[3]{##3}%
124     \renewcommand\includegraphics[2] [] {\app@ig[#1]{##2}}%
125     \input{#2}%
126 \end{postscript}}

```


`\mathfig` For Mathematica's MathPSfrag output

```
127 \newcommand\mathfig[2] [] {%
128   \begin{postscript}
129     \input{#2-psfrag}%
130     \includegraphics[#1]{#2-psfrag}%
131   \end{postscript}}
```

`\psfragfig` EPS graphics via psfrag. Include your psfrag commands in the files *<document>-psfrag.tex* and/or *<figname>-psfrag.tex*, where *<document>* is the filename of the main document and *<figname>* is the filename of the graphics inserted.

```
132 \newcommand\psfragfig[2] [] {%
133   \@ifnextchar[
134     {\app@psfragfig[#1]{#2}}
135     {\app@psfragfig[#1]{#2} []}}
136 \def\app@psfragfig[#1]#2[#3] {%
137   \begin{postscript}
138     \InputIfFileExists{#2-psfrag}{-}{-}%
139     #3
140     \includegraphics[#1]{#2}%
141   \end{postscript}}
```

Finally, input any psfrag commands associated with the document:

```
142 \InputIfFileExists{\jobname-psfrag}{-}{-}
```